Colin Pham

Professor Brooks

Foundations of AI

23 September 2024

Assignment 2: Search - Question 2 Sub Problems

Modify your search code so that it instead solves three subproblems: moveToSample, removeSample, and returnToCharger. You can do this by changing the start state and goal test. How does this change the number of states generated?

Depth Limited Search will not be used as I want to see how long it will truly take Depth First Search to find the solution. Running the entire search from the start gives me

28 Breadth First Search States and 41 Depth First Search States

It looks like Breadth First Search eventually queued the solution while Depth First Search looked in too many depths that the solution was nowhere in.

I decided to divide the problem up into smaller problems. First, for my move_to_sample goal, I had the search algorithms stop as soon as the location of the rover was at the sample site. Running it gives me

3 Breadth First Search States and 5 Depth First Search States

It looks like both searches found the solution fairly quickly since the solution is extremely close to the starting point. Depth First Searc

Next, I had the remove_sample goal, where it starts from the sample location and had it complete when the rover extracts the sample from the site (uses the tool). Running it gives me

6 Breadth First Search States and 6 Depth First Search States

Both searches completed the move_to_sample goal in 2 steps meaning that they both still needed some time to pick up and use the tool. Again, since the goal was so close to its starting point, both searches completed with around the same efficiency

Finally, I ran the return_to_charger goal where the search algorithms had to return to the charging station after using the tool to extract the sample. Running it gives me

17 Breadth First Search States and 15 Depth First Search States

The starting point and goal are a lot further this time. This time, both searches have to worry about going from the sample site, picking up the sample, dropping off the sample, and then moving to the battery. The goal was apparently close enough for both searches to complete around the same time. Whether or not it is holding a tool is negligible for my implementation, but changing the goal to make it so the robot can either carry a tool or sample (but not both) is possible.

In conclusion, for both searches, if I were to just divide the entire problem into smaller ones and run them individually, I would end up with less states generated. However, in the case of the Mars Rover, I would rely on breadth first searches. Having Depth First Search run problem by problem is a good way to solve its inefficiency problem seen when running the whole thing altogether